Comparison of Knowledge Graph Representations for Consumer Scenarios

Ana Iglesias-Molina[1](✉), Kian Ahrabian[2], Filip Ilievski[2], Jay Pujara[2], and Oscar Corcho[1]

[1] Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain
{ana.iglesiasm,oscar.corcho}@upm.es

[2] Information Sciences Institute, University of Southern California, Marina del Rey, CA, USA
{ahrabian,ilievski,jpujara}@isi.edu

Abstract. Knowledge graphs have been widely adopted across organizations and research domains, fueling applications that span interactive browsing to large-scale analysis and data science. One design decision in knowledge graph deployment is choosing a representation that optimally supports the application's consumers. Currently, however, there is no consensus on which representations best support each consumer scenario. In this work, we analyze the fitness of popular knowledge graph representations for three consumer scenarios: knowledge exploration, systematic querying, and graph completion. We compare the accessibility for knowledge exploration through a user study with dedicated browsing interfaces and query endpoints. We assess systematic querying with SPARQL in terms of time and query complexity on both synthetic and real-world datasets. We measure the impact of various representations on the popular graph completion task by training graph embedding models per representation. We experiment with four representations: Standard Reification, N-Ary Relationships, Wikidata qualifiers, and RDF-star. We find that Qualifiers and RDF-star are better suited to support use cases of knowledge exploration and systematic querying, while Standard Reification models perform most consistently for embedding model inference tasks but may become cumbersome for users. With this study, we aim to provide novel insights into the relevance of the representation choice and its impact on common knowledge graph consumption scenarios.

Keywords: Knowledge Graphs · Knowledge Representation · User Study · Graph Completion

1 Introduction

The growth of the knowledge graph (KG) user base has triggered the emergence of new representational requirements. While RDF is the traditional and standard model for KG representation, alternative models such as property graphs [25], the Wikidata model [34], and RDF-star [12] have also become recently popular. The promise of these alternative and complementary representation models is that they can provide more flexibility to address certain use cases, such as statement annotation, for which RDF-based representations are not straightforward [17]. While the plurality of knowledge representation (KR) models provides the means to address a wider range of possibilities in consumer scenarios, there is currently no consensus nor sufficient empirical evidence on which representations are most suitable for different KG consumer tasks [16].

Previous studies comparing knowledge representations have focused primarily on query performance [2,6,14,26,28] and graph interoperability [3,4]. For this scenario, the representations need to ensure efficiency to minimize performance time. However, applications relating to exploration by end users and machine learning over KGs have not been taken into account [16,22]. Knowledge exploration scenarios, e.g., browsing, are impacted by representational choices, and therefore the selected representations should reduce the cognitive load and user expertise needed to explore, access, and acquire knowledge. Similarly, many embedding models-based tasks such as knowledge graph completion [1,29,37] require adequate representations to maximize the performance of the models on downstream predictive tasks.

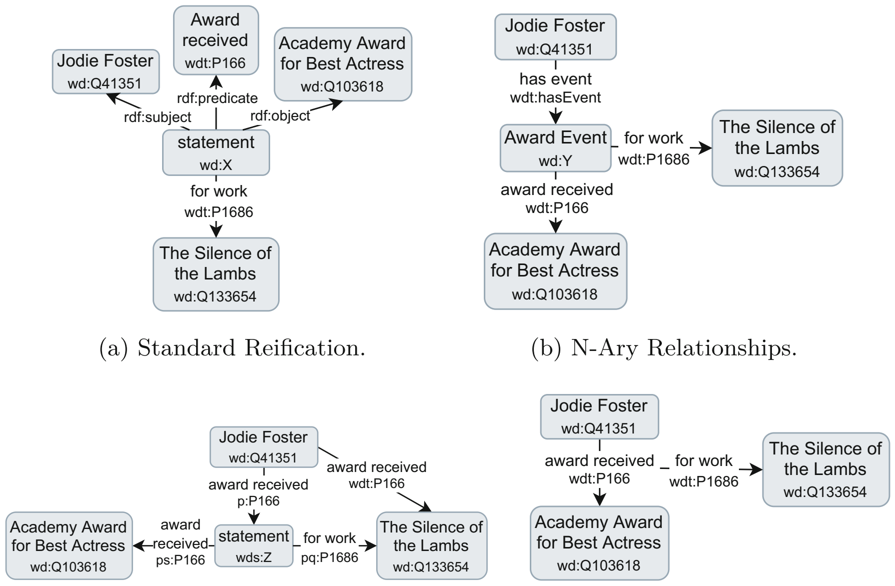
In this paper, we address the research question: **How do different knowledge representation models impact common KG consumer scenarios?** Based on three complementary KG consumer tasks (knowledge exploration, systematic querying, and knowledge graph completion), we define four concrete questions: **(RQ1)** Which representations facilitate faster knowledge exploration and acquisition? **(RQ2)** Are certain representations more intuitive for building accurate queries efficiently? **(RQ3)** How do representational query patterns impact query evaluation time? **(RQ4)** How do different representations affect the performance of KG embedding models for a KG completion task?

We investigate these research questions by assessing the fitness of four popular KR approaches: Standard Reification, N-Ary Relationships, Wikidata qualifiers, and RDF-Star, for the needs of the abovementioned scenarios. First, to understand user preferences in knowledge exploration tasks, we run a user study where participants interact with a web browser interface and a query endpoint to determine the representation that improves knowledge acquisition for real-world questions. Then, to assess the differential performance of the representations, we test several queries using synthetic and real-world data. Lastly, to estimate the impact on KG embedding model performance, we train and evaluate a selection of these models for the KG completion task with different representations.

The rest of the paper is structured as follows: Sect. 2 introduces the four representation models. Section 3 describes the datasets used in the evaluation. The experimental setup and evaluation are organized by scenario, for knowledge exploration in Sect. 4, systematic querying in Sect. 5, and knowledge graph embedding models in Sect. 6. Section 7 reviews related work. The conclusions and limitations are discussed in Sect. 8.

2 Knowledge Representation Models

In this section, we describe different representation models that can be used for statement annotation, i.e., making statements about statements, a challenge that has motivated the development of several different KG representation approaches [7,12,24,26,27]. Figure 1 illustrates instances of these models for the main statement *Jodie Foster received the Academy Award for Best Actress* annotated with the additional statement *for the work The "Silence of the Lambs."*



(a) Standard Reification. (b) N-Ary Relationships.

(c) Qualifiers. (d) RDF-star.

Fig. 1. Representation models shown as RDF graphs: (a) Standard Reification, (b) N-Ary Relationships, (c) Qualifiers (Wikidata model) and (d) RDF-star.

Standard Reification. [24] (Fig. 1a) explicitly declares a resource to denote an `rdf:Statement`. This statement has `rdf:subject`, `rdf:predicate`, and `rdf:object` attached to it and can be further annotated with additional statements. The resource is typically a blank node; we simplify the encoding using a Wikidata Item identifier (shown as `wd:X` in the figure for brevity).

N-Ary Relationships. [27] (Fig. 1b) converts a relationship into an instance that describes the relation, which can have attached both the main object and additional statements. This representation is widely used in ontology engineering as an ontology design pattern [10].

The Wikidata Model. [7] (Fig. 1c) is organized around the notion of `Items`. An `Item` is the equivalent of either a Class or an Instance in RDF and is described with labels, descriptions, aliases, statements, and site links. Statements represent triples (comprised of `item-property-value`) and can be further enriched with qualifiers and references. Qualifiers are `property-value` pairs that attach additional information to the triple. From this point onward, we refer to this representation as "Qualifiers".

RDF-Star. [12] (Fig. 1d) extends RDF to introduce triple reification in a compact manner. It introduces the notion of triple recursiveness with `Quoted Triples`, which can be used as subjects and/or objects of other triples. The RDF-star graph shown in Fig. 1d is represented in RDF as follows: <<`wd:Q41351 wd:P166 wd:Q103618`>> `wdt:P1686 wd:Q133654`.

Table 1. Number of triples ($\times 10^6$) for the WD-AMC dataset and the REF benchmark in the analysed representations.

	Qualifiers	RDF-star	N-Ary Rel.	Std. Reif.
WD-AMC	30.917	5.700	35.653	51.266
REF benchmark	268.965	61.033	140.521	175.592

3 Datasets

WD-AMC. (Wikidata - Actors, Movies, and Characters) is a novel subset of Wikidata introduced in this paper to simulate a real-world scenario of manageable size. To this end, we first extract manually a list of actors, characters, and movies present in the questions provided by the WebQuestionSP [35] and GoogleAI benchmarks.[1] Then, we sample WD-AMC by taking all properties and values of the main Wikidata statements for the entities in this list, along with their qualifier properties and values. The WD-AMC subset and all its variants are created using the Knowledge Graph Toolkit (KGTK) [21].
For the query evaluation performance scenario, we use the **REF benchmark** [28], which is proposed to compare different reifications providing the Biomedical Knowledge Repository (BKR) dataset [30] in three representations: Standard Reification, Singleton Property, and RDF-star. We extend the available representations in REF to include Qualifiers and N-Ary Relationships approaches. Table 1 presents the number of triples of both datasets in each representation. We make all datasets and their corresponding queries available online [19,20].

[1] https://ai.google.com/research/NaturalQuestions/dataset.

4 Knowledge Exploration Scenario

We define *knowledge exploration* as the process of interactively discovering and obtaining information available in knowledge graphs. We distinguish and study two knowledge exploration scenarios by asking users to (i) interact with a user-friendly interface, and (ii) build queries to systematically access the KG. We measure the time and accuracy of the participant responses for both scenarios.

4.1 Experimental Setup

User Study Setup. We conduct a user study composed of two tasks. The *browser interaction task* consists of answering 5 natural language questions by looking for the information in Wikidata via a KGTK browser interface. The answer should be provided as a Wikidata identifier (QXXXX). The *endpoint interaction task* consists of building SPARQL queries for the same natural language questions answered in the previous task, providing a machine-executable query as a response. In this task, participants can build and test the query in a triplestore provided for them. For both tasks, answers are submitted in free-text boxes, with no predefined options to choose from. Participants are provided with representative example responses in order to ensure that they submit the queries in a useful format for the posterior evaluation. We measure the time spent solving each query and the accuracy of the responses.

Table 2. Set 1 of questions used in the user study with their corresponding identifier (ID) and from which QA benchmark they were extracted (Source).

ID	Questions	Source
Q1	Cast of the 2005 version of Pride and Prejudice	GoogleAI
Q2	What character did Natalie Portman play in The Phantom Menace?	WebQSP
Q3	Who won the academy award for best actor in 2020?	GoogleAI
Q4	Who is the Australian actress in Orange Is The New Black?	GoogleAI
Q5	How many movies have Woody Harrelson and Bill Murray been in together?	GoogleAI

Data. For both tasks, we use the WD-AMC dataset described in Sect. 3. For the *browser interaction task*, we adapt the dataset to create three instances (one per representation) of the KGTK Browser,[2] an adaptive user interface similar to Wikidata. For the *endpoint interaction task*, we generate the corresponding RDF graphs and upload them to a triplestore. We do not include RDF-star explicitly since its visualization in the first task is equivalent to the Qualifier model representation. All resources are accessible online for the participants.[3,4]

[2] https://github.com/usc-isi-i2/kgtk-browser/.
[3] https://kgtk.isi.edu/krhc1/, https://kgtk.isi.edu/krhc2/, https://kgtk.isi.edu/krhc3/.
[4] https://setting1.krhc.linkeddata.es/sparql.

Queries. We prepare 4 sets of 5 queries, extracting them from the QA benchmarks WebQuestionsSP [35] and GoogleAI(see footnote 1), which contain real questions from users. One set of queries is presented in Table 2 in natural language. Each of the 4 sets contains variants of the same 5 queries to minimize participants' risk of copying, by altering specific elements in the questions, e.g., years, movies; while maintaining the structure.[5] These 4 versions are applied to the three representations, resulting in 12 different sets of queries. We map these query sets to 12 groups of participants. Participants are provided with the same set of natural language questions for completing both tasks.

Participants. The user study is carried out with 45 students of a Master's level course on Knowledge Graphs. All students have similar backgrounds, are enrolled in a University Computer Science or Data Science program, and have learned about the KG representations in the course. The students first sign up for a voluntary assignment and are randomly divided into 12 groups, 4 groups per representation and each of them with a different query set. Then, they are sent a Google questionnaire with representation description, instructions, questions for both tasks, and text boxes for the answers. These groups contain: 16 participants for Qualifiers, 16 for N-Ary Relationships, and 13 for Standard Reification.

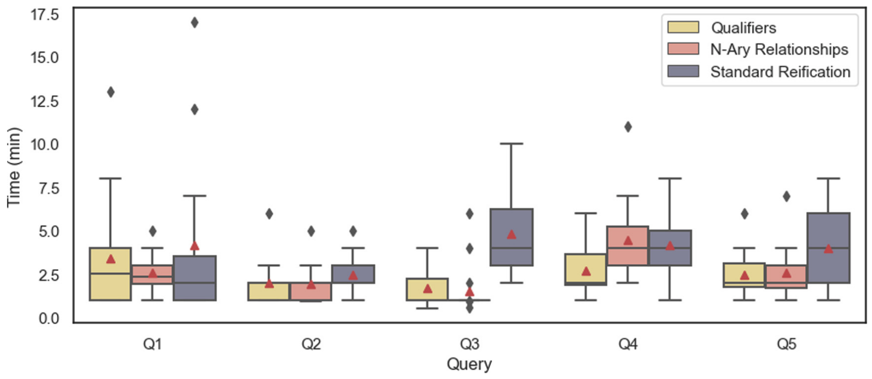


Fig. 2. Measured time that participants spent for retrieving the answers to the questions proposed in the *browser interaction task*.

Metrics. We analyze the results using ANOVA and t-test for the response time measurements to look for significant differences among representation approaches per query. Both ANOVA and t-test are used under the assumptions of (i) normality, (ii) sameness of variance, (iii) data independence, and (iv) variable continuity. ANOVA is first used for the three variables (Qualifiers, N-Ary Relationships and Standard Reification). Then, the t-test is used to test pairs of representations per query. To test the significance of accuracy, we use

[5] https://github.com/oeg-upm/kg-scenarios-eval/blob/main/WD-AMC/README.md.

the chi-squared test of independence and look into whether the accuracy and representation variables are independent. It is used under the following assumptions: (i) the variables are categorical, (ii) all observations are independent, (iii) cells in the contingency table are mutually exclusive, and (iv) the expected value of cells in the contingency table should be 5 or greater in at least 80% of the cells.

4.2 Results

Browser Interaction Results. Figure 2 shows participants' time spent finding the answer in the browser interface for each query. We observe that the results for the three representations are overall similar, with significant differences in individual queries. The ANOVA test shows significant differences between the representation models (p-value< 0.05) for queries Q3-5 (Table 3). Our t-tests for pairs of representation models confirm the results of ANOVA for Q1-2, showing no significant differences. For Q3 and Q5, the measured time is significantly higher for Standard Reification, while for Q4, Qualifiers take significantly less time compared to the other representations. Thus, we observe that the time spent answering questions with the Standard Reification is significantly higher for Q3-5; and the average time in Q1 and Q2 is slightly (but not significantly) higher, which makes this representation less fit than the other two for this task. To obtain the accuracy of the responses, we measure the proportion of correct responses retrieved. Nearly all of the received answers are correct. Only in one query with N-Ary Relationships and Standard Reification, a wrong answer is submitted. We run a chi-squared test per query, and the results show no significant differences among the approaches in terms of accuracy.

Table 3. P-values of ANOVA and t-test for the time that participants spent for retrieving the answers to the questions proposed in *browser interaction task*. The significant values (p-value < 0.05) are highlighted in **bold**.

		Q1	Q2	Q3	Q4	Q5
ANOVA		0.437	0.429	**7.87E-06**	**0.034**	**0.047**
t-test	Qualifiers - N-Ary Rel.	0.323	0.964	0.715	**0.013**	0.817
	Qualifiers - Std. Reif.	0.621	0.289	**1.42E-04**	**0.034**	**0.034**
	N-Ary Rel. - Std. Reif.	0.216	0.220	**1.80E-04**	0.735	**0.064**

Answering **RQ1**, we conclude that, while participants can find the correct answers with any of the three representations, answering questions via knowledge exploration takes longer with the Standard Reification representational model. This finding is intuitive, as Standard Reification divides a triple into three triples, where only the object is the relevant element. The information is thus scattered and does not follow the “natural” direction of relationships between the elements. For instance, for answering Q3 "Who won the academy award for best actor in 2020?", in Qualifiers and N-Ary Relationships the information stems from the *Academy Award for Best Actor* (Q103916) node using the *winner* property (P1346); while for Standard Reification, the statement holds this information in separate relations: <`wd:Statement rdf:subject wd:Q103916; rdf:predicate wdt:P1346`>. Thus, the answer can be only accessed by referencing from the statement node, rather than directly as for the other representations.

Endpoint Interaction Results. Figure 3 shows the distribution of time spent on building the SPARQL queries and Table 4 shows the accuracy of the results obtained when running the SPARQL queries submitted by the participants. In terms of time, we note small variations among the three representations, which are not significant according to the ANOVA test. In terms of accuracy, the results for this task show a higher portion of errors compared to the *browser interaction task*, and they vary highly among approaches and queries. The largest differences are observed for the queries Q1 and Q5, where the Qualifiers model performs the best (accuracy of 1 and 0.71) and the Standard Reification model has an accuracy of 0.36. In such queries, Standard Reification requires a `UNION` clause

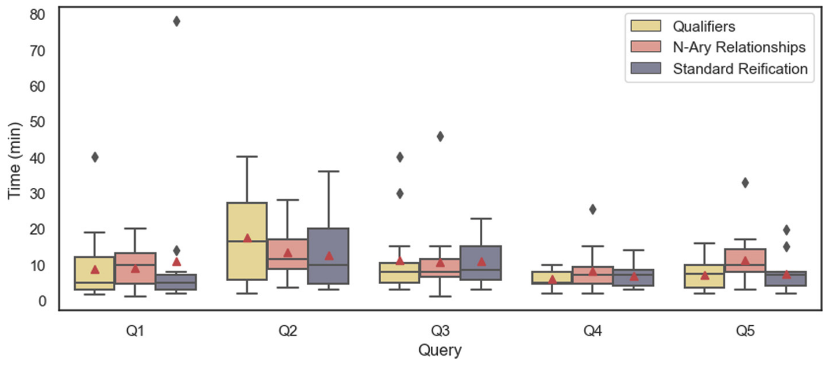


Fig. 3. Measured time that participants spent for building the SPARQL queries corresponding to *endpoint interaction task*.

Table 4. Proportion of correct responses returned by the SPARQL queries built in the *endpoint interaction task*. The highest accuracy per query is highlighted in **bold**, and the lowest is underlined. The p-values of the chi-squared test are also shown. Significant values (p-value <0.05) are marked with *.

	Q1	Q2	Q3	Q4	Q5	Average
Qualifiers	**1.000**	0.786	0.786	0.786	**0.714**	0.814
N-Ary Rel.	0.938	**0.875**	0.813	**0.938**	0.563	**0.825**
Std. Reif.	0.364	0.818	**0.909**	0.818	0.364	0.655
Chi-squared test	4.340E-05*	0.874	0.756	0.532	0.133	0.022*

to retrieve the complete set of results, which is not needed for the other representations, and thus, increases the relative complexity of the correct query. The results of the other three queries are relatively close between the three representation models, with Qualifiers performing the worst on all of them. However, on average Standard Reification produces the lowest accuracy, while N-Ary Relationships the highest. To test the significance of the accuracy results, we apply chi-squared tests Table 4), showing significant values for Q1 and on average. Thus, the accuracy of results is in general dependent on the representation.

Addressing **RQ2**, we observe that all three representations perform similarly in terms of the time it takes to interact with a SPARQL endpoint. However, for queries with a higher complexity, Standard Reification is more error-prone, as it requires additional clauses (e.g., `UNION`) to retrieve the complete set of results. Curiously, these results are similar to those for the *browser interaction task* in RQ1, in the sense that Standard Reification fares the worst among the three models, while Qualifiers and N-Ary Relationships perform alternatively best depending on the query. Yet, the granular performance on individual queries and metrics overlaps only partially: in the *browser interaction task*, the gap is observed in terms of time and affects queries Q3–Q5; while in the *endpoint interaction task*, it is manifested in terms of accuracy for the queries Q1 and Q5. These findings provide initial insights into the suitability of different representation models for knowledge exploration. We leave a more systematic comparison between queries in terms of their properties for future work.

Table 5. Characteristics of the WD-AMC queries in terms of the number of triple patterns and SPARQL clauses used per query. The RDF-star queries with quoted triples are marked with "*", while "^{SR}" only applies to Std. Reif. The number of checkmarcks (✓) indicates the number of times the clause appears.

		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
#TP	**Qualifiers**	1	3	3	3	2	3	3	3	2	6
	RDF-Star	1	1*	1*	3	2	1*	3	3	2	2*
	N-Ary Rel.	2	3	3	6	4	3	3	4	4	6
	Std. Reif.	3	3	4	5	8	4	4	2	2	8
Clauses	**FILTER**	–	–	✓	–	–	–	–	✓✓	–	–
	FUNCTION	–	–	✓	–	–	–	–	✓✓	–	–
	UNION	–	–	–	–	✓^{SR}	–	–	–	–	–
	OPTIONAL	–	–	–	–	–	–	✓	–	–	–

Table 6. Characteristics of queries of the REF benchmark regarding the number of triple patterns and SPARQL clauses used per query. The number of checkmarcks (✓) indicates the number of times the clause appears. GT stands for the *greater than* operator.

		A-Q1	A-Q2	A-Q3	A-Q4	B-Q1	B-Q2	B-Q3	F-Q1	F-Q2	F-Q3	F-Q4	F-Q5
#TP	**Qualifiers**	3	2	4	4	3	6	9	3	4	6	9	6
	RDF-Star	1	1	2	2	1	2	3	1	2	2	3	2
	N-Ary Rel.	4	4	5	5	4	8	12	4	5	7	12	8
	Std. Reif.	4	5	5	6	4	10	15	5	6	10	15	10
Clauses	**COUNT**	–	–	✓	✓	–	–	–	–	–	–	–	–
	GROUP BY	–	–	✓	–	–	–	–	–	–	–	–	–
	FILTER	–	–	–	✓	–	–	–	✓	✓	✓	✓	✓✓
	FUNCTION	–	–	–	✓	–	–	–	✓	✓	–	–	–
	Operator (GT)	–	–	–	–	–	–	–	–	–	✓	✓	✓✓

5 Systematic Querying Scenario

The *systematic querying* scenario refers to the assessment of information retrieval efficiency with diverse query loads and structures. We analyze the differential behaviour of diverse series of queries over realistic and synthetic data for each representation, measuring its performance time.

5.1 Experimental Setup

Data. We use both datasets presented in Sect. 3. The WD-AMC dataset is used for analysing the behaviour with real-world queries in real-world data. We reuse the REF benchmark to validate our results with their previously reported analysis [28], and extend the resources to test and analyse two additional representations, Qualifiers and N-Ary Relationships.

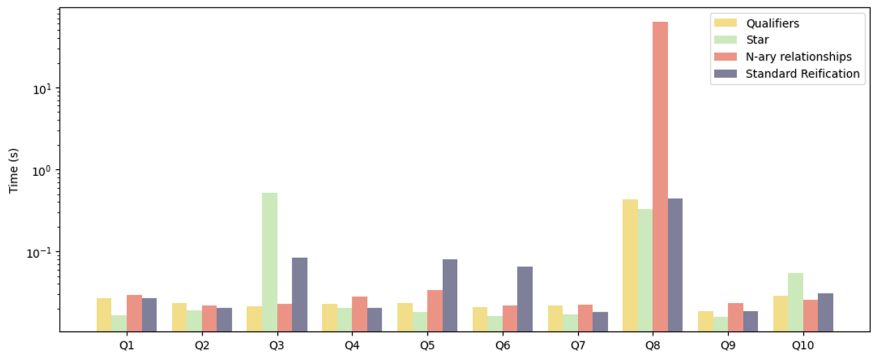


Fig. 4. Measured query evaluation time for the SPARQL WD-AMC dataset.

Queries. For WD-AMC, we use 10 series of queries. Each series is comprised of 20 variants of one query extracted from the QA benchmarks WebQuestionsSP [35] and GoogleAI[3], which contain real questions from users made over Google. We use the first queries shown in Table 2, and we introduce five additional queries, extracted from the same QA benchmarks, to study a wider variety of query structures. The query variants are created by altering specific elements in the query (e.g., years, actors, movies, characters) while maintaining the triple patterns intact. Their characteristics are shown in Table 5. They are selected to apply to the patterns that differentiate the representations variants of WD-AMC. We note that, being extracted from real questions and not designed by us, not all queries require the use of the reification solution in all representations. This is especially remarkable for RDF-star, which is highlighted in the table, and directly affects the results described below. Still, all queries are different across the four representations.

The REF benchmark contains 12 queries divided into three series per approach (A, B, and F). These series contain queries with variable length, complexity, and computationally expensive clauses such as `COUNT` or `FILTER`, or operators such as *greater than* (Table 6). For more details on the queries, we refer to [30] for series A, [26] for series B, and [28] for series F.

Implementation. Previous studies [14] show no significant differences between the proposed representations across different triplestores. For that reason, we only use Jena Fuseki, which is an open-source implementation that can process all four representations. Both datasets were uploaded to a Jena Fuseki v4.6.1 triplestore running on a single-node VM with a 4-cores CPU and 16GB of main memory. To measure the efficiency of each representation, we measure the evaluation time while assessing the query complexity. Each set of queries for both datasets is run in warm mode three times, and the average is shown as a result.

5.2 Results

Results on WD-AMC. Figure 4 shows the average query evaluation time on the WD-AMC benchmark. As all queries are naturally asked by users, they are typically not computationally expensive, returning results in less than 100 ms. Q8 is the only exception, requiring more time as it contains additional clauses (2 functions and 2 `FILTER`; cf. Table 5). In this case, N-Ary Relationships take longer (around 1 min) as it contains more triple patterns than the other representations. Most of the remaining queries show similar times, with Qualifiers providing the quickest response on average[7]. In some queries, Standard Reification needs a higher number of triple patterns compared to the other representations, resulting in higher response times. For Q5, in addition, this representation requires the `UNION` clause to provide the complete set of results, a clause that is not needed in the other representations. RDF-star performs the best for nearly all queries, which may be due to having the smallest size and lowest number of triple patterns per query. The queries for which this representation shows a significantly increased response time involve expensive clauses or joins.

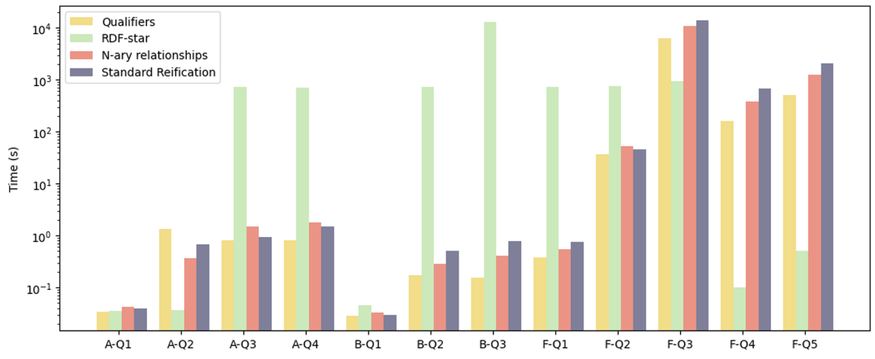


Fig. 5. Measured query evaluation time for the REF benchmark.

Results on the REF Benchmark. Figure 5 depicts the results obtained for the evaluation using the REF benchmark. RDF-star and Standard Reification show similar results to those reported by Orlandi et al. [28]. We observe that for Qualifiers, N-Ary Relationships, and Standard Reification, the measured times follow a similar pattern in query evaluation time for all queries in contrast to RDF-star. Qualifiers obtain the best results for almost all queries, presenting a total execution time reduced by half compared to the other representations.[6] The measured response times for RDF-star show a completely different behavior. Most of the queries for which this representation presents an increased time response involve joins. Meanwhile, RDF-star is not affected as much as the other representations by the use of operators such as *greater than* (Table 6).

Addressing **RQ3**, query performance on both datasets is sensitive to the choice of representation. The Qualifiers representation shows the quickest results for demanding queries on average. Together with N-Ary Relationships and Standard Reification, these representations are usually less differentially affected by a particular clause or pattern compared to RDF-star, as these three show similar behaviour per query. RDF-star stands out for good performance for simple queries (i.e., with one quoted triple) and logical operators such as *greater than*, but it becomes highly inefficient when joins and clauses like `FILTER` or `COUNT` are involved. RDF-star introduces quoted triples, a new element in the syntax that implies different processing by triplestores compared to the other representations, and makes it the most susceptible to change between queries.

6 Knowledge Graph Embedding Scenario

Machine learning applications over graphs have relied on the idea of *knowledge graph embedding* (KGE). KGE provides a mechanism to map the nodes and edges in a KG into a high-dimensional vector space. The resulting vector representations are then used for a variety of tasks, including link prediction, node

[6] https://github.com/oeg-upm/kg-scenarios-eval/blob/main/REF-Benchmark/README.md.

classification, graph classification, and entity resolution. In this section, we evaluate popular KGE methods on the task of *knowledge graph completion* (KGC), which addresses the sparsity of a KG by predicting an object node given a subject node and a relation. We study the impact of the different KG representations on KGC performance in terms of mean reciprocal rank (MRR) and hits@K.

6.1 Experimental Setup

Data. We use the WD-AMC dataset described in Sect. 3. RDF-star is not used in this evaluation because the models that use this representation as their input are not fir for large-scale data. To create the evaluation data, we first extract all the triples containing a statement node and an object node from each representation. In the example shown in Fig. 1, this would yield the triples <`wd:X, rdf:object, wd:Q103618`>, <`wd:Y, wdt:P166, wd:Q103618`>, and <`wds:Z, ps:P166, wd:Q10361`>. Next, for each representation, we randomly sample 10% of the respective triples into a test and validation set, and combine the rest with the remaining triples of each representation to create the training sets. After this procedure, we end up with the following number of triples for our experiments: 1) validation set: 189,839, 2) test set: 189,839, 3) qualifiers train set: 5,128,864, 4) n-ary train set: 3,610,155, and 5) standard reification train set: 5,440,482.

Methodology. KGE models learn a mathematical relationship between entities and relationships that allow them to produce a likelihood score for an arbitrary triple. The most common architecture for these models is the encoder-decoder framework [11]. Traditionally, the encoder consists of learnable shallow embeddings $(\mathcal{E}, \mathcal{R})$ that map each entity or relation to a high-dimensional vector, and the decoder is a function that takes in the high-dimensional representations and produces the likelihood score. Formally, these models could be defined as $\mathcal{L}(s, r, o) = \psi(\mathcal{E}(s), \mathcal{R}(r), \mathcal{E}(o))$, where (s, r, o) is the given arbitrary triple, $\mathcal{E}$ is the shallow embedding for entities, $\mathcal{R}$ is the shallow embedding for relations, ψ is the decoder function, and $\mathcal{L}$ is the produced likelihood score. For our experiments, we use three of the most common KGE models with publicly accessible large-scale implementations [23,36], namely RotatE [31], ComplEx [32], and TransE [5]. We exclude graph neural network models from this study as none had any publicly accessible implementation that could operate on the scale of our dataset in a reasonable time [18].

Implementation. In the evaluation phase, we compare each positive sample with 4,096 negative samples and report the object prediction results on the test set. Moreover, to mitigate the effect of random negative sampling in the evaluation phase, we run each experiment five times and report the best result. To make a fair comparison, we fix the following hyperparameters: *training steps* = 300 k and *batch size* = 1024. As for the rest of the hyperparameters, they were tuned on the following ranges: *embedding dimension* $\in \{50, 100, 200\}$ *learning rate* $\in \{0.003, 0.01, 0.03, 0.1\}$, *regularization coefficient*

Table 7. KGC results for three KGE models. We bold the best result for a model.

	RotatE				ComplEx				TransE			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
Qualifiers	**0.716**	0.678	**0.744**	**0.778**	0.426	0.364	0.453	0.546	0.599	0.518	0.658	0.739
N-Ary Rel.	0.714	**0.683**	0.736	0.765	0.261	0.232	0.277	0.310	**0.721**	**0.678**	**0.752**	**0.793**
Std. Reif.	0.697	0.653	0.726	0.773	**0.518**	**0.468**	**0.542**	**0.613**	0.622	0.567	0.655	0.725

$\in \{1e{-}4, 1e{-}5, 1e{-}6, 1e{-}7, 1e{-}8, 1e{-}9\}$, *negative samples size* $\in \{512, 1024\}$, *adversarial temperature* $\in \{0.5, 1.0\}$, and *gamma* $\in \{18, 12, 9, 6, 3, 2, 1\}$.

6.2 Results

Table 7 presents the experimental results of the models above on the KGC task. We observe that each model has a specific best-performing representation with significant performance differences compared to the other representations. However, the performance gap decreases as the expressive power of the model increases, indicating that the sheer expressive power of the models could potentially overcome representational disadvantages. From the perspective of representations, we observe that although two of the top three best-performing models use N-Ary Relationships, Standard Reification is the best representation on average. This phenomenon showcases the potential downfall of randomly mix-and-matching models with representations.

Regarding **RQ4**, our experimental results showcase the importance of associating models with a suitable representation. Failing to do so may lead to degraded performance, as evident from the results obtained on N-Ary Relationships with ComplEx. Throughout our experiments, no single representation consistently achieves the highest performance; however, we observe that a model with superior expressive power, i.e., RotatE, can overcome potential representational shortcomings and perform consistently well over all representations.

7 Related Work

Multiple studies have assessed the efficiency of different representations in terms of data management efficiency with computationally expensive queries, measuring metrics such as execution time, storage size, or number of triples. The Singleton Properties [26] model is compared in its inception with Standard Reification [24] using two series of queries of increasing complexity. Hernández et al. [14] test Wikidata represented in N-Ary Relationships, Standard Reification, Singleton Properties and Named Graphs over multiple triplestores. They remark the processing difficulties for Singleton Properties due to the high number of created properties. In a follow-up work, Hernández et al. [15] investigate how Wikidata in the same representations as their previous study performed on two SPARQL triplestores (Virtuoso and Blazegraph), a relational database (PostgreSQL), and a graph database (Neo4J), reporting that Virtuoso performed best among the data stores. Frey et al. [9] extends this work's evaluation to the DBpedia dataset and its representations to include their proposal Companion Property and Blazegraph's Reification Done Right,[7] which is currently known as RDF-star) This work employs several data stores, highlighting which representation performs best in each store. More recently, the REF-benchmark [28] is proposed to evaluate different reification approaches with the inclusion of RDF-star, providing a version of the Biomedical Knowledge Repository (BKR) dataset [30] and three series of queries for each representation that can be applied to different triplestores. Additionally, there are studies that compare Property Graphs and RDF with regards to query performance [2,6] and model interoperability [3,4], highlighting the advances of Property Graphs.

Hence, studies so far focus on the behavior of multiple representations over triplestores, and occasionally relational databases or other graph databases. To the best of our knowledge, there is no comprehensive evaluation of KG representations across consumer scenarios beyond triplestores and studies of query efficiency, a gap filled in by the extensive evaluation of different representations and scenarios in this paper. Yet, we include the systematic querying scenario in our work to (i) validate our experimental setup with the REF benchmark, obtaining similar results to ones previously reported, (ii) enrich it with two new representations (N-Ary Relationships and Wikidata qualifiers), and (iii) mimic a real-world scenario with real data and queries.

8 Discussion and Conclusions

8.1 Summary of Results

In this paper, we assessed the fitness of different knowledge graph representations in three consumer scenarios: knowledge exploration, systematic querying, and KG embedding. While no single representation model was optimal for all scenarios, we found significant differences for particular consumer scenarios, summarized in Table 8. We can extract the following conclusions from our study: (i) Standard Reification is the least suitable for users. Its anti-intuitive structure results time-consuming to navigate with, and it introduces additional complexity to retrieve correct and complete information. (ii) RDF-star still needs improved support in all studies scenarios, as it is underway of becoming part of the RDF 1.2 specification [13]. At the moment, it is risky to use it in high-demanding scenarios. (iii) Qualifiers obtain steadily better results for retrieving results in high-demanding querying scenarios. Despite being restricted to Wikidata at the moment, its representation could be considered to be adopted in more knowledge graphs. (iv) Analysing and understanding how each embedding model works is key to select a representation for graph completion (and hence, additional embedding-based tasks). While for the other scenarios all representations showed acceptable behaviour, here the decision is critical. (v) Promoting the use of interfaces such as browsers highly improves the user experience in knowledge

[7] https://github.com/blazegraph/database/wiki/Reification_Done_Right.

Table 8. Summary of the fitness for each representation evaluated in the studied scenarios, where ✓ means suitable, ∽ is acceptable, X is avoidable and * indicates that the value is not tested but equivalent to Qualifiers.

	User interaction	Simple graphs and queries	Large graphs and demanding queries	Graph completion
Qualifiers	✓	✓	✓	✓ (RotatE)
RDF-star	✓*	✓	∽	–
N-Ary Rel.	✓	✓	∽	✓ (TransE)
Std. Reif.	X	✓	∽	✓ (ComplEx)

exploration. These interfaces help mask the representation complexity and differences, which directly influences the adoption and usability of semantic resources, an aspect usually overlooked. (vi) Despite the lack of a good-for-all solution, promoting interoperability among representations when knowledge graphs are consumed for very different purposes is potentially useful.

8.2 Limitations

Knowledge graphs are built for reuse across different applications, and their fitness for these applications depends on representational choices. Our paper provided insights into the fitness of four representations for three diverse consumption scenarios. We reflect on three key design decisions and point to future extensions of this study that can increase the significance of its findings.

1. Representational choices - Our study considered a subset of representations, prioritized for their popularity in prior work. Namely, we selected two RDF-based representations (the Standard Reification proposed in the first RDF Recommendation and the widely used N-Ary Relationships); the Wikidata model, and RDF-star. Other RDF-based representations (e.g. Singleton Properties) and property graphs were out of scope of this paper, but their inclusion in follow-up work is valuable. The choice of representations directly influences the selection of particular techniques and tools, as a representation may be limited to a single technology and cannot be processed by others. This prevents evaluating all existing representations under the exact same conditions. To ensure fair evaluation conditions, property graphs were not included in this work. Hence, further work is needed to include additional relevant representations while ensuring that differences in performance are due to the representation and not the underlying implementation.

2. Task choices - The tasks studied in this paper were derived by surveying popular KG tasks from recent research and associating them with three consumer scenarios. For knowledge exploration, we selected two representative tasks, yet, it remains to be seen whether our findings will generalize to other exploratory tasks, e.g., graph navigation visualization. For systematic querying, we extend previous studies to address a real-world scenario. The generality of this scenario can be increased with a complete real-world dataset instead of a subset with increasingly complex queries and with additional query languages (e.g., Cypher [8] or

Linked Data Fragments [33]). For KG embedding, a vibrant research community has explored a vast space of learning methods, applications, and benchmarks. We limit the task to KG completion and investigate simpler, widely used models to focus on a common real-world scenario. We believe these choices balance the likely trade-offs in many deployed settings, where institutional knowledge and dataset size preclude the adoption of more advanced techniques. Alternatives in the task space (e.g., node classification or entity resolution) and the model space, where graph neural network-based architectures can more directly represent complex graph structures, are promising extensions we hope to investigate in future work.

3. Participant sample - The remaining limitation refers to the sample of participants in the knowledge exploration user study. For this first study, we chose Master students because of their short and comparable experience with knowledge graph representations and tasks. We considered them a better sample than colleague practitioners, who may be already biased toward the representations and technologies they use. As a next step of this work, we plan to include a more varied sample of participants, i.e., to include industry experts, junior students, academics, and software developers.

Supplemental Material Statement: Datasets are available from Zenodo [19], queries and supplementary resources from GitHub [20].

Acknowledgement. We would like to thank Pedro Szekely for the inception and preliminar conceptualization of this work; Daniel Garijo and Edna Ruckhaus for the feedback provided for improving this work. This work was partially funded by the project *Knowledge Spaces* (Grant PID2020-118274RB-I00 funded by MCIN/AEI/10.130 39/501100011033).

References

1. Alivanistos, D., Berrendorf, M., Cochez, M., Galkin, M.: Query embedding on hyper-relational knowledge graphs. arXiv preprint: arXiv:2106.08166 (2021)
2. Alocci, D., Mariethoz, J., Horlacher, O., Bolleman, J.T., Campbell, M.P., Lisacek, F.: Property graph vs RDF triple store: a comparison on glycan substructure search. PLoS ONE **10**(12), e0144578 (2015)
3. Angles, R., Thakkar, H., Tomaszuk, D.: RDF and property graphs interoperability: status and issues. AMW **2369**, 1–11 (2019)
4. Angles, R., Thakkar, H., Tomaszuk, D.: Mapping RDF databases to property graph databases. IEEE Access **8**, 86091–86110 (2020)
5. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
6. Das, S., Srinivasan, J., Perry, M., Chong, E.I., Banerjee, J.: A tale of two graphs: property graphs as RDF in oracle. In: EDBT, pp. 762–773 (2014)
7. Erxleben, F., Günther, M., Krötzsch, M., Mendez, J., Vrandečić, D.: Introducing Wikidata to the linked data web. In: Mika, P., et al. (eds.) ISWC 2014. LNCS, vol. 8796, pp. 50–65. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11964-9_4

8. Francis, N., et al.: Cypher: an evolving query language for property graphs. In: Proceedings of the 2018 International Conference on Management of Data, pp. 1433–1445 (2018)
9. Frey, J., Müller, K., Hellmann, S., Rahm, E., Vidal, M.E.: Evaluation of metadata representations in RDF stores. Semantic Web **10**(2), 205–229 (2019)
10. Gangemi, A., Presutti, V.: A multi-dimensional comparison of ontology design patterns for representing n-ary relations. In: van Emde Boas, P., Groen, F.C.A., Italiano, G.F., Nawrocki, J., Sack, H. (eds.) SOFSEM 2013. LNCS, vol. 7741, pp. 86–105. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-35843-2_8
11. Hamilton, W.L.: Graph Representation Learning. Synthesis Lectures on Artifical Intelligence and Machine Learning, vol. 14, no. 3, pp. 1–159 (2020)
12. Hartig, O.: Foundations of RDF* and SPARQL* (An Alternative Approach to Statement-Level Metadata in RDF). In: Proceedings of the 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web. CEUR Workshop Proceedings, vol. 1912 (2017)
13. Hartig, O., Champin, P.A., Kellog, G.: RDF 1.2 concepts and abstract syntax. W3C Working Draft, World Wide Web Consortium (2023). https://www.w3.org/TR/rdf12-concepts/
14. Hernández, D., Hogan, A., Krötzsch, M.: Reifying RDF: what works well with Wikidata? vol. 1457, pp. 32–47 (2015)
15. Hernández, D., Hogan, A., Riveros, C., Rojas, C., Zerega, E.: Querying Wikidata: comparing SPARQL, relational and graph databases. In: Groth, P., et al. (eds.) ISWC 2016. LNCS, vol. 9982, pp. 88–103. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46547-0_10
16. Hogan, A.: The semantic web: two decades on. Semantic Web **11**(1), 169–185 (2020)
17. Hogan, A., et al.: Knowledge graphs. ACM Comput. Surv. (CSUR) **54**(4), 1–37 (2021)
18. Hu, W., Fey, M., Ren, H., Nakata, M., Dong, Y., Leskovec, J.: OGB-LSC: a large-scale challenge for machine learning on graphs. arXiv preprint: arXiv:2103.09430 (2021)
19. Iglesias-Molina, A.: Comparison of knowledge graph representations for consumer scenarios - datasets. https://doi.org/10.5281/zenodo.7443836 (2023)
20. Iglesias-Molina, A.: oeg-upm/kg-scenarios-eval: v1.0.0. https://github.com/oeg-upm/kg-scenarios-eval, https://doi.org/10.5281/zenodo.8179156 (2023)
21. Ilievski, F., et al.: KGTK: a toolkit for large knowledge graph manipulation and analysis. In: Pan, J.Z., et al. (eds.) ISWC 2020. LNCS, vol. 12507, pp. 278–293. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62466-8_18
22. Karger, D.R.: The semantic web and end users: what's wrong and how to fix it. IEEE Internet Comput. **18**(6), 64–70 (2014)
23. Lerer, A.: PyTorch-BigGraph: a large scale graph embedding system. Proc. Mach. Learn. Syst. **1**, 120–131 (2019)
24. Manola, F., Miller, E.: RDF primer. W3C Recommendation, World Wide Web Consortium (W3C) (2004). https://www.w3.org/TR/rdf-primer/
25. Miller, J.J.: Graph database applications and concepts with Neo4j. In: Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA, vol. 2324 (2013)
26. Nguyen, V., Bodenreider, O., Sheth, A.: Don't like RDF reification? Making statements about statements using singleton property. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 759–770 (2014)

27. Noy, N., Rector, A.: Defining N-ary relations on the semantic web: use with individuals. Technical report, W3C (2006). https://www.w3.org/TR/swbp-n-aryRelations/
28. Orlandi, F., Graux, D., O'Sullivan, D.: Benchmarking RDF metadata representations: reification, singleton property and RDF. In: 2021 IEEE 15th International Conference on Semantic Computing (ICSC), pp. 233–240. IEEE (2021)
29. Ren, H., et al.: SMORE: knowledge graph completion and multi-hop reasoning in massive knowledge graphs. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1472–1482 (2022)
30. Sahoo, S.S., Bodenreider, O., Hitzler, P., Sheth, A., Thirunarayan, K.: Provenance context entity (PaCE): scalable provenance tracking for scientific RDF data. In: Gertz, M., Ludäscher, B. (eds.) SSDBM 2010. LNCS, vol. 6187, pp. 461–470. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13818-8_32
31. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: RotatE: knowledge graph embedding by relational rotation in complex space. In: International Conference on Learning Representations (2019). https://openreview.net/forum?id=HkgEQnRqYQ
32. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning, pp. 2071–2080. PMLR (2016)
33. Verborgh, R., Vander Sande, M., Colpaert, P., Coppens, S., Mannens, E., Van de Walle, R.: Web-scale querying through linked data fragments. In: LDOW (2014)
34. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
35. Yih, W.T., Richardson, M., Meek, C., Chang, M.W., Suh, J.: The value of semantic parse labeling for knowledge base question answering. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 201–206 (2016)
36. Zheng, D., et al.: DGL-KE: training knowledge graph embeddings at scale. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 739–748 (2020)
37. Zhu, Z., Galkin, M., Zhang, Z., Tang, J.: Neural-symbolic models for logical queries on knowledge graphs. In: International Conference on Machine Learning, pp. 27454–27478. PMLR (2022)